One-stage and Dual-heuristic Particle Swarm Optimization for Virtual Network Embedding

An Song
School of Computer Science and Technology
South China University of Technology
Guangzhou, China
safe.song@qq.com

Wei-Neng Chen
School of Computer Science and Technology
South China University of Technology
Guangzhou, China
cwnraul634@aliyun.com

Xiao-Min Hu
School of Computers
Guangdong University of Technology
Guangzhou, China
xmhu@ieee.org

***Abstract*—Virtual network embedding (VNE) is the key technology in network virtualization and has been proven NP-hard. The purpose of VNE is to find the optimal mapping of virtual nodes and links, and minimize the utilization of resources. However, many particle swarm optimization approaches to VNE separate VNE into two independent subproblems (i.e., node mapping and link mapping) and ignore the coordination between node mapping and link mapping. In this paper, a one-stage and dual-heuristic particle swarm optimization (DH-PSO) is devised to solve VNE. To coordinate node mapping and link mapping, firstly, DH-PSO updates positions of particles step by step, and nodes and links are mapped in one stage. Secondly, DH-PSO devises the dual-heuristic strategy to further improve the optimizing capability. The first heuristic strategy is to construct a candidate set and the second strategy is to find the best solution from the candidate set. Hence, not only the network resources but the network paths are taken into account to construct solutions. DH-PSO can be combined with different two-stage approaches to become one-stage. DH-PSO is experimentally studied on different instances. The experimental results verify that the proposed DH-PSO is promising.**

***Index Terms*—Virtual network embedding, particle swarm optimization, metaheuristics.**

I. Introduction

VIRTUAL network embedding (VNE) is the critical technology in network virtualization and has been intensively studied in recent years [1, 2]. VNE aims to map virtual networks (VNs) requested by customers to a substrate network (SN) owned by Infrastructure providers. In this way, heterogeneous applications are allowed to coexist on the same hardware, which is considered as a promising way to overcome Internet ossification [3-5].

This work was supported in part by the Key Project of Science and Technology Innovation 2030 supported by the Ministry of Science and Technology of China (Grant No. 2018AAA0101300), in part by the National Natural Science Foundation of China under Grants 61976093 and 61772142, in part by the Science and Technology Plan Project of Guangdong Province 2018B050502006, and in part by Guangdong Natural Science Foundation Research Team 2018B030312003. Corresponding author: Wei-Neng Chen (cwnraul634@aliyun.com)

As extensions of subgraph isomorphism [6, 7], VNE problems consist of two subproblems, the mapping from virtual nodes to substrate nodes (i.e., VNoM) and the mapping from virtual links to substrate links (i.e., VLiM). During the embedding process, resource constraints should be satisfied, such as the CPU requirements of virtual nodes and the bandwidth requirements of virtual links. Due to these constraints, VNE is an intractable discrete combinatorial optimization problem and has been proven NP-hard [1, 8].

As VNE is important and challenging, different algorithms are proposed to solve VNE, including heuristic algorithms and metaheuristic algorithms [9-13]. Among these, particle swarm optimization (PSO) [14] has shown its superiority due to easy implementation and steady convergence. Several PSO-based approaches are devised, such as unified enhanced PSO (UEPSO) [15], PSO with random walk (RWPSO) [16] and distributed VNE with set-based PSO [17]. Most PSO-based approaches deal with VNoM and VLiM independently, called two-stage mechanism. The two-stage mechanism first finds the mapping of all virtual nodes (i.e., first stage) and then find the mapping of all virtual links (i.e., second stage). As shown in Fig.1 (a), the nodes a, b and c are all mapped, and then the links (a, c) and (c, b) are mapped. The two-stage mechanism is a straightforward way to solve VNE. However, the two-stage mechanism has the problem that the feasible mapping of virtual

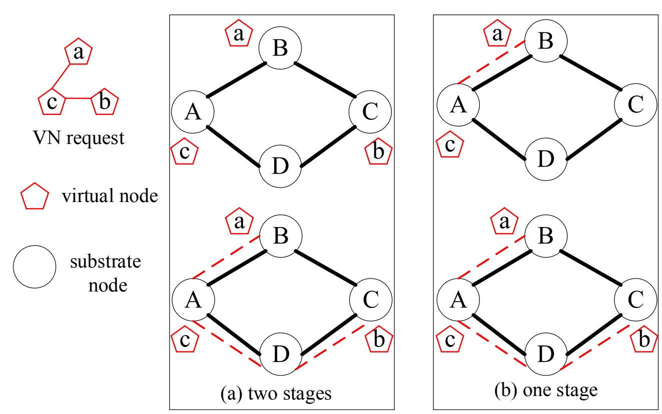

Fig.1. The comparison between two-stage and one-stage mechanism.

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

links may not exist after VNoM due to resource constraints, which means the two-stage mechanism ignores the cooperation between VNoM and VLiM.

On the contrary, the one-stage mechanism finds feasible VLiM after mapping a single virtual node, which can coordinate VNoM and VLiM together. As shown in Fig.2 (b), the nodes a and c, and the link (a, c) are mapped simultaneously. The effectiveness of one-stage mechanism has been verified in heuristic approaches, such as vnmFlib [6] and RW-BFS [18]. How to adopt the one-stage mechanism in PSO-based approaches to VNE is critical to improving the optimizing capability of them.

In this paper, we propose the one-stage and dual-heuristic PSO (DH-PSO) to solve VNE problems. Firstly, DH-PSO adopts the one-stage mechanism and updates positions of particles step by step. Every time a virtual node is mapped to a substrate node, DH-PSO checks the mapping of virtual links adjacent to the virtual node. Consequently, VNoM and VLiM are coordinated in one stage to reduce invalid construction of candidate solutions. Secondly, DH-PSO devises the dual heuristic strategy to further improve the optimizing capability of DH-PSO. The first heuristic strategy uses the roulette wheel method to construct a set of candidate nodes. The second heuristic strategy uses the path information of networks to find the "best" node from the candidate set. The dual heuristic strategy includes both information of network resources and information of network paths. Hence, adjacent virtual nodes can be mapped to short paths on substrate networks.

DH-PSO is more than a one-stage approach to VNE, and it can be combined with different two-stage PSO of VNE to become one-stage. To verify the effectiveness and generality of proposed DH-PSO, the representative two-stage PSO approach, RWPSO [16] and UEPSO [15], is combined with DH-PSO, namely DH-UEPSO and DH-RWPSO. The compared algorithms are experimentally studied in 100-node substrate networks and 20-node virtual networks, which are commonly used network sizes in VNE field.

The rest of this paper is organized as follows. Section II introduces the background of VNE, including the problem definition, related work and PSO. The detailed elaboration of the proposed DH-PSO is presented in Section III. In Section IV, experiments are conducted to verify the efficiency and effectiveness of DH-PSO. Finally, Section V concludes this paper.

II. BACKGROUND

A. Problem Definition

VNE aims to find the mapping from virtual networks (VN) to a substrate network (SN), satisfying the resource constraints. The SN and VN are both defined on weighted undirected graphs. Let $\boldsymbol{G}_s=(\boldsymbol{N}_s, \boldsymbol{L}_s)$ be a substrate network, where $\boldsymbol{N}_s$ represents the set of substrate nodes and $\boldsymbol{L}_s$ represents the set of substrate links. Let $\boldsymbol{G}_v=(\boldsymbol{N}_v, \boldsymbol{L}_v)$ be a virtual network. where $\boldsymbol{N}_v$ represents the set of virtual nodes and $\boldsymbol{L}_v$ represents the set of substrate links. Nodes and links in $\boldsymbol{G}_s$ and $\boldsymbol{G}_v$ have resource attributes. Generally, a substrate node $n\in\boldsymbol{N}_s$ (or virtual node $n\in\boldsymbol{N}_v$) is associated with CPU resources, denoted as $\mathrm{CPU}_s(n)$ (or $\mathrm{CPU}_v(n)$). Here the subscript "s" and "v" represent substrate and virtual networks, respectively. Similarly, a substrate link $l\in\boldsymbol{L}_s$ (or virtual link $l\in\boldsymbol{L}_v$) is associated with bandwidth resources, denoted as $\mathrm{BW}_s(l)$ (or $\mathrm{BW}_v(l)$). As shown in Fig.2, the numbers on virtual and substrate nodes represent the CPU resources. The numbers on virtual and substrate links represent the bandwidth resources.

The VNE problem consists of two subproblems, virtual node mapping (VNoM) F_N: $\boldsymbol{N}_v \longrightarrow \boldsymbol{N}_s'$ and virtual link mapping (VLiM) F_L: $\boldsymbol{L}_v \rightarrow \boldsymbol{P}_s'$. Here $\boldsymbol{N}_s'\subset\boldsymbol{N}_s$ and $\boldsymbol{P}_s'\subset\boldsymbol{P}_s$, where $\boldsymbol{P}_s$ is the set of paths in $\boldsymbol{N}_s$. As shown in Fig.2, the VNoM is {a→B, b→C, c→A} and the VLiM is {(a, c)→(B, A), (c, b) →(A, D, C)}. During the VNoM and VLiM, the following constraints Eq.(1)-(4) should be satisfied,

$$\forall u\in N_v, \sum_{i\in N_s} x_i^u = 1 \tag{1}$$

$$\forall i\in N_s, \sum_{u\in N_v} x_i^u \le 1 \tag{2}$$

$$\forall u\in N_v, \forall i\in N_s, x_i^u \times \mathrm{CPU}_v(u) \le \mathrm{CPU}_s(i) \tag{3}$$

$$\forall l_{ij}\in L_s, \forall l_{uv}\in L_v, f_{ij}^{uv} \times \mathrm{BW}_v(l_{uv}) \le \mathrm{BW}_s(l_{ij}) \tag{4}$$

x_i^u and f_{ij}^{uv} are binary variables. x_i^u=1 means virtual node v is mapped to substrate node i and x_i^u=0 means the opposite. f_{ij}^{uv} =1 means virtual link l_{uv} is mapped to the substrate link l_{ij} and f_{ij}^{uv}=0 means the opposite. Eq.(1) represents all virtual nodes should be mapped. Eq.(2) represents a substrate node is allowed to host one virtual node at most. Eq.(3) represents virtual nodes should be mapped to the substrate nodes with more CPU resources than the virtual nodes. Eq.(4) represents virtual links should be mapped to the substrate links with more bandwidth resources than the virtual links.

The objective of VNE is to minimize the resource occupation, including CPU and bandwidth resources, formulated as

$$\min \sum_{u\in N_v} \mathrm{CPU}_v(u) + \sum_{l_{uv}\in L_v} \sum_{l_{ij}\in L_s} f_{ij}^{uv} \times \mathrm{BW}_v(l_{uv}) \tag{5}$$

For example, the objective value in Fig.2 is evaluated as (10+15+20)+(10×1+15×2)=85.

B. Algorithms of VNE

As a NP-hard problem, different algorithms are proposed to

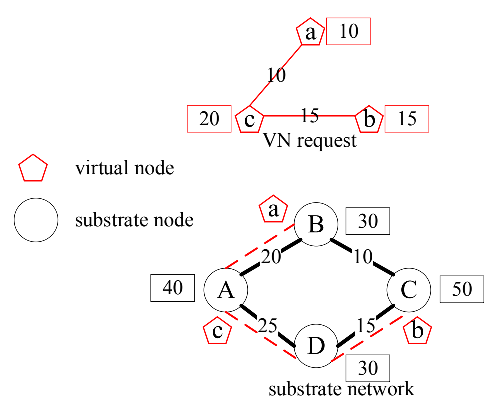

Fig.2. An example of embedding a virtual network on a substrate network.

solve VNE, including heuristic algorithms and metaheuristic algorithms.

Heuristic algorithms use problem-dependent information to search for solutions. Derived from the subgraph isomorphism detection (SID) problem, Lischka and Karl [6] proposed vnmFlib to solve VNE in one stage. A VN is viewed as a subgraph of the SN with resource constraints. vnmFlib uses backtracking search to find the feasible node mapping and link mapping simultaneously. Cheng *et al*. [16, 18] take topology attributes of networks into account and proposed the node ranking metric inspired by the PageRank algorithm. Based on the node ranking metric, two heuristic algorithms, RW-BFS and RW-MaxMatch, are proposed. RW-BFS uses breadth-first search to construct solutions, which belongs to one-stage approach. On the contrary, RW-MaxMatch directly assigns virtual nodes with high ranks to substrate nodes with high ranks, which belongs to two-stage approaches. To coordinate VNoM and VLiM, Chouwdhury *et al*. [8, 19] formulate VNE as a mixed integer program. Then the integer constraints are relaxed to obtain a linear program, and D-ViNE (deterministic strategy) and R-ViNE (randomized strategy) are proposed.

Heuristic approaches to VNE can quickly find feasible solutions but they usually get stuck in local optima [1]. Metaheuristic approaches are able to focus on global search. Zhang *et al*. [15] and Cheng *et al*. [16] devise unified enhanced PSO (UEPSO) and PSO with random walk (RWPSO) to solve VNE. UEPSO and RWPSO redefine the velocity and position updating in the VNE space. Thus particles can fly to better positions guided by velocities. D. Palanikkumar and S. Priya [20] propose Ant Colony based [21] Graph Theory, which breaks down networks as topological sequences. Node and links are mapped jointly in conjunction to eliminate infeasible mappings. Song *et al*. [22] focus on large-scale situations and propose evolutionary algorithms with overlapped decomposition (ODEA). A large VNE problem is decomposed into overlapped subproblems and the competitive strategy is devised to cooperate subproblems. Different algorithms can be combined with ODEA and they are improved in large-scale network situations.

C. Particle Swarm Optimization

PSO is one of the most popular evolutionary algorithms and it was first proposed by Kennedy and Eberhart in 1995 [14]. PSO is inspired by the social behavior of animals, such as birds flocking and fish schooling. PSO has been extensively studied and has been widely used in continuous and discrete optimization problems [23-26]. PSO consists of a population of particles and each particle maintains two vectors, a position vector and a velocity vector. Particles learn from the best solution found by the population so far and the best solution historically found by their own to update velocities. Then particles can fly to better positions guided by updated velocities. For the ith particle (i = 1, 2,…, NP), its velocity and position are denoted as $\boldsymbol{V}_i=(v_i^1, v_i^2,\dots, v_i^n)$ and $\boldsymbol{X}_i=(x_i^1, x_i^2,\dots, x_i^n)$, respectively. For the jth dimension of particle i, the updating rules of the velocity and position are as follows:

$$v_i^j \leftarrow w\times v_i^j + c_1 r_1^j \times (pbest_i^j - x_i^j) + c_2 r_2^j \times (gbest^j - x_i^j) \quad (6)$$

$$x_i^j \leftarrow x_i^j + v_i^j \quad (7)$$

where w is the inertia weight, $pbest_i$ and $gbest$ are the best solution found by the particle itself and the population, respectively. c_1 and c_2 are two parameters to indicate the importance of $pbest_i$ and $gbest$. r_1^j and r_2^j are two random numbers uniformly distributed in [0,1].

III. Proposed DH-PSO

To coordinate the relation between VNoM and VLiM, DH-PSO is proposed to solve VNE in one stage. The fundamental contribution of DH-PSO is the step-by-step position updating and the dual heuristic strategy. In this section, we introduce the encoding scheme, velocity updating and position updating in detail.

A. Encoding Scheme

Like original PSO, particles in DH-PSO also maintain position vectors and velocity vectors. For the ith particle, the position represents the mapping of virtual nodes, denoted as $\boldsymbol{X}_i = (x_i^1,\dots,x_i^d,\dots,x_i^D)$ where x_i^d is the substrate node to host the dth virtual node and D is the number of virtual nodes. For example in Fig. 2, the position is encoded as (B, C, A).

Velocities in DH-PSO indicate the evolving direction of particles. The velocity $\boldsymbol{V}_i$ for the ith particle is a probability vector, denoted as $\boldsymbol{V}_i=(v_i^1,\dots v_i^d,\dots, v_i^D)$ where $v_i^d \in [0,1]$. v_i^d represents the probability of unchanging the mapping of the dth virtual node. v_i^d=0 represents the mapping of the dth virtual node should be changed while v_i^d=1 means the mapping of the dth virtual node maintains unchanged.

B. Velocity Updating

The velocity updating in original PSO is defined on the continuous space (i.e., Eq.(6)). To adapt to the discrete space in VNE, the mathematical operations in Eq.(6) should be redefined. Inspired by the velocity updating in UEPSO and RWPSO, the rule of velocity updating for the ith particle on dimension d is redefined as follows,

$$v_i^d = wv_i^d \oplus c_1(pbest_i^d \ominus x_i^d) \oplus c_2(gbest^d \ominus x_i^d) \quad (8)$$

where,

$$v_i^{'} = pbest_i^d \ominus x_i^d = \begin{cases} 0 & \text{if } pbest_i^d \neq x_i^d \\ 1 & \text{if } pbest_i^d = x_i^d \end{cases} \quad (9)$$

$$wv_i^d \oplus c_1 v_i^{'} = \min\{wv_i^d + c_1 v_i^{'}, 1\} \quad (10)$$

Here the parameters w, c_1 and c_2 have the same role in Eq.(6). The subtraction between two position vectors is defined as Eq.(9). The purpose of subtraction is to find the difference between the current position and a better position. If a virtual node in the current position is mapped to the same substrate node as that in *pbest*, the result of subtraction is 1, which promotes the probability of unchanging in Eq. (10). Otherwise, the result is 0, which will increase the probability of changing the node mapping. The addition operator between probability vectors is defined in Eq.(10). The probabilities in vectors are directly added and the addition results cannot exceed one.

For example, given $\boldsymbol{pbest_i}$ = (2,3,4) and $\boldsymbol{X_i}$ = (2,1,5), $\boldsymbol{pbest_i} \ominus \boldsymbol{X_i}$ =(1,0,0). Given $\boldsymbol{V_i}$ = (0.9,0.4,0.1), 0.3×$\boldsymbol{V_i}$=(0.27,0.12,0.03) 0.3×$\boldsymbol{V_i}$ $\oplus$ 0.7×($\boldsymbol{pbest_i} \ominus \boldsymbol{X_i}$)=(0.97,0.12,0.03). The result of

Algorithm 1:virtual node mapping

input: The size of candidate set CS, the virtual node v, the velocity ***vol*** and the old mapped substrate node s.

output:the substrate node s'.

```
0:  generate random number rand in [0,1];
1:  if vol[v] >= rand//keep the virtual node mapping unchanged
2:      s' = s;
3:  else//update the virtual node mapping
4:      candidate set candi_set = {};
5:      for i= 1 : CS
6:          select a candidate substrate node n with roulette wheel selection method;// the first heuristics
7:          candi_set = candi_set ∪ {n};
8:      end for
9:      select the substrate node hn from candi_set with the least SP value; // the second heuristics
10:     s' = hn;
11: end if
12: find the link mapping of s' and check the feasibility;
13: if s' is invalid
14:     randomly select a node rn from the substrate network;
15:     s' = rn;
16:     find the link mapping of s' and check the feasibility;
17:     if s' is invalid
18:         reinitialize the particle;
19:     end if
20: end if
```

addition " $\oplus$ " means the probability of unchanging the mapping of virtual nodes. For the first virtual node in the addition result, the probability of unchanging is 0.97. As for the second virtual node, the probability of unchanging is 0.12, which means the probability of changing is 1-0.12=0.88.

C. Position Updating

After velocity updating, particles can fly to better positions guided by velocities. The position updating in DH-PSO adopts the one-stage mechanism and the node mapping in positions is updated step by step. This is the major difference between DH-PSO and other two-stage PSO algorithms. The pseudo code of virtual node mapping (VNoM) is presented in Algorithm 1. The VNoM consists of two parts. The first part is to learn from the velocity (lines 0-11 in Algorithm 1) and the second part is the one-stage mapping (lines 12-20 in Algorithm 1).

According to the updated velocities, promising elements of VNoM are assigned with high probabilities in velocities and they should be kept unchanged in the next generation. We generate a random number *rand* ($rand \in [0,1]$) to identify the quality of VNoM. Given virtual node v, if its probability in the velocity is larger than *rand*, the mapping of v is good enough and is kept unchanged. Otherwise, the mapping of v should be improved and we need to find another substrate node to host v.

1) Dual-heuristic Strategy

We devise the dual-heuristic strategy to find appropriate substrate nodes. The first heuristic strategy is to construct a candidate set of substrate nodes with roulette wheel selection. Note that different attributes of nodes can be used to evaluate the probability of selecting substrate nodes. Here we adopt the *NR* metric used in UEPSO [15] as an example, which is evaluated as follows for substrate node $n \in \boldsymbol{N}_s$,

$$NR(n) = \mathrm{CPU}_s(n) \times \sum_{l \in L(n)} \mathrm{BW}_s(l) \tag{11}$$

where $L(n)$ is the set of adjacent links of node n. The probability of selecting substrate node n is evaluated as,

$$\mathrm{prob}(n) = \frac{\mathrm{NR}(n)}{\sum_{m \in N_s'} \mathrm{NR}(m)} \tag{12}$$

where $N_s' \subset N_s$ is the set of substrate nodes with adequate CPU resources. For networks with many nodes or sufficient resources, the size of N_s' is quiet large. Hence, one time of roulette wheel selection may ignore some substrate nodes with great potential. To find more promising substrate nodes, we repeat the roulette wheel selection CS (CS =2,3,4,…) times to construct a candidate set ***candi_set*** (i.e., |***candi_set***|=CS) of substrate nodes, which is the result of the first heuristic strategy (lines 5-8 in Algorithm 1).

The second heuristic strategy needs to find the best element in ***candi_set***. We define the metric *SP* to evaluate the quality of elements in ***candi_set*** to host virtual node v. *SP* is based on the shortest paths of substrate networks. For the substrate node $n \in$ ***candi_set***, $SP(v, n)$ is defined as follows,

$$SP(v,n) = \sum_{m \in \boldsymbol{M}(v)} dist(n,m) \tag{13}$$

where $dist(n, m)$ represents the length of the shortest path between n and m in the SN. $\boldsymbol{M}(v)$ is a set of substrate nodes who host virtual nodes adjacent to v. The metric *SP* measures the distance between the candidate substrate node n and mapped substrate nodes. Substrate node n with smaller *SP* is nearer to the mapped substrate nodes and thus less bandwidth resources are likely to consume. Hence, the substrate node with the least *SP* is selected from ***candi_set*** (line 9 in Algorithm 1).

2) One-stage Mapping

Many evolutionary algorithms adopt the two-stage mechanism to solve VNE, which ignores the cooperation of node mapping and link mapping. DH-PSO adopts one-stage mechanism to coordinate node mapping and link mapping in one stage, which is the second part of Algorithm 1.

After learning from the velocity, a candidate substrate node s' is selected to host virtual node v. s' can be selected from the old mapping result (line 2 in Algorithm 1) or selected with the duel-heuristic strategy (line 9 in Algorithm 1). In both situations, the feasibility of s' should be checked, which is the major difference between one stage and two stages.

To check the feasibility of s', we need to find that whether the mapping of virtual links connecting to s' exists or not. We use the fast link mapping algorithm [15, 17] to quickly find link mapping. To map a virtual link l, the substrate links with less bandwidth resources than l are temporarily removed. Second, the shortest path algorithm is executed to find the path connecting s' and another mapped substrate node. If a feasible link mapping cannot be found, s' is invalid and we randomly select another free substrate node from the substrate network. If s' is still invalid, this particle will be discarded and reinitialized.

Every time we select a substrate node s' for node mapping, DH-PSO will check and find the corresponding link mapping. After all virtual nodes are mapped, all virtual links are mapped simultaneously. During the node mapping, each node is able to mapped twice at most. Thus it is more probable to construct feasible solutions, which is the strength of one-stage approaches.

TABLE I
Comparison Results of Costs for Compared Algorithms

		DH-UEPSO	UEPSO	DH-RWPSO	RWPSO
ins01	mean	**4.479E+03**	4.797E+03	**4.381E+03**	4.667E+03
	std	1.067E+02	6.349E+01	9.697E+01	6.522E+01
	p-value	0.000		0.000	
ins02	mean	**4.338E+03**	4.656E+03	**4.234E+03**	4.520E+03
	std	6.175E+01	5.744E+01	1.041E+02	7.944E+01
	p-value	0.000		0.000	
ins03	mean	**4.487E+03**	4.830E+03	**4.414E+03**	4.683E+03
	std	8.535E+01	6.961E+01	8.766E+01	6.268E+01
	p-value	0.000		0.000	
ins04	mean	**4.573E+03**	4.860E+03	**4.482E+03**	4.753E+03
	std	1.123E+02	1.147E+02	1.012E+02	7.255E+01
	p-value	0.000		0.000	
ins05	mean	**5.016E+03**	5.315E+03	**4.961E+03**	5.196E+03
	std	8.832E+01	9.441E+01	1.105E+02	5.512E+01
	p-value	0.000		0.000	
ins06	mean	**5.027E+03**	5.338E+03	**4.913E+03**	5.228E+03
	std	1.130E+02	8.112E+01	1.076E+02	6.390E+01
	p-value	0.000		0.000	
ins07	mean	**5.238E+03**	5.506E+03	**5.129E+03**	5.402E+03
	std	7.047E+01	9.872E+01	9.017E+01	5.738E+01
	p-value	0.000		0.000	
ins08	mean	**5.363E+03**	5.606E+03	**5.249E+03**	5.481E+03
	std	1.223E+02	8.526E+01	9.386E+01	6.495E+01
	p-value	0.000		0.000	
ins09	mean	**4.822E+03**	5.121E+03	**4.687E+03**	4.978E+03
	std	1.040E+02	6.009E+01	8.660E+01	6.768E+01
	p-value	0.000		0.000	
ins10	mean	**4.657E+03**	4.963E+03	**4.524E+03**	4.820E+03
	std	1.053E+02	6.132E+01	1.156E+02	5.939E+01
	p-value	0.000		0.000	

IV. Experimental Studies

In this section, we conduct experiments to investigate the performance of DH-PSO. Firstly, the experimental environment is introduced, including parameter settings and compared algorithms. Then the compared algorithms are tested in different scenarios.

A. Experimental Environment

Similar to previous studies [15, 16, 27], the size of substrate networks is 100 and the average connectivity rate is fixed at 10%. The size of virtual networks is 200 and the average connectivity rate is fixed at 10%. For fair comparisons, the topologies of substrate and virtual networks are generated by the Georgia Tech Internetwork Topology Models (GT-ITM) tool [28]. The bandwidth and CPU resources are uniformly distributed in [50, 100] for SNs and distributed in [1, 50] for VNs.

DH-PSO can adopt different metrics of substrate nodes as the first heuristic strategy. Two kinds of metrics are combined with DH-PSO, the *NR* metric used in UEPSO (as stated in subsection III. C) and the node rank with PageRank used in RW-PSO. The corresponding algorithms are denoted as DH-UEPSO and DH-RWPSO. UEPSO and RW-PSO are representative two-stage approaches. DH-UEPSO and DH-RWPSO can be viewed as the one-stage versions of them. UEPSO is compared with DH-UEPSO, and RW-PSO is compared with DH-RWPSO to verify the effectiveness of DH-PSO.

The size of populations is set to 40 and the max number of generations is set to 100 for all compared algorithms. The parameters w, c_1 and c_2 are set to 0.1, 0.2 and 0.7, respectively. The size of candidate set *CS* (line 5 in Algorithm 1) is set to 6. For fair comparisons, all algorithms are implemented in Java and the programs are executed on computers with Intel(R) Core(TM) i5-4590 CPU at 3.30GHz. The operating system is Linux and the JDK version is 1.8.

B. Comparison on Costs

The objective of VNE is to minimize the costs of mapping VNs, which is defined in Eq. (5). The comparison results with regard to costs are presented in Table I. Since the compared algorithms are stochastic approaches, 30 independent runs are executed for each algorithm on each instance and thus their average performance can be studied. We conduct Wilcoxon rank sum test at significance level 0.05 to examine whether compared results are significantly different. Based on the p-value of Wilcoxon rank sum test, the best results are highlighted in bold.

From Table I, it can be seen that DH-UEPSO can significantly outperform UEPSO, and DH-RWPSO can significantly outperform RWPSO in all instances. These results verify that DH-PSO is effective and can improve the performance of different two-stage approaches. On the one hand, DH-PSO updates positions step by step, which can coordinate node mapping and link mapping in one stage. On the contrary, UEPSO and RWPSO are two-stage approaches, which ignore the cooperation between node mapping and link mapping. On the other hand, DH-PSO devises the dual heuristic strategy to further improve the performance. UEPSO and RWPSO only use roulette wheel selection to select candidate substrate nodes. If there are many available nodes to select, the roulette wheel selection is difficult to find the appropriate one. Hence, DH-PSO uses roulette wheel selection to construct a set of candidate substrate nodes as the first heuristic strategy. Then DH-PSO uses the shortest path information to find the best one from the candidate set as the second strategy. In this way, adjacent virtual nodes can be closely mapped to substrate nodes, and thus the costs (CPU and bandwidth resources) are reduced.

Fig.3 presents the converging curves of compared algorithms. The x-axis is the number of generations (100 generations are the maximum) and the y-axis is the objective value. The dashed lines represent the original two-stage approaches and the solid lines represent DH-PSO based approaches. From Fig.3, it can be observed that DH-PSO based approaches can converge to lower objective values in all instances, which verifies the effectiveness of DH-PSO. Another interesting phenomenon is that RWPSO based approaches are better than UEPSO based approaches. This is because RWPSO uses network topology attributes in roulette wheel selection, which can contain more information than only node resource attributes in UEPSO.

C. Comparison on Invalid Solutions

Essentially, VNE is a kind of discrete combinatorial optimization problem with constraints. The constraint rules in VNE include CPU and bandwidth resource constraints. If the resources in substrate networks become congested, some mapping results may violate the constraint rules (i.e., Eq.(1)-Eq.(4)). As particle swarm optimization algorithms stochastically find solutions, it is very common that a valid solution may become invalid after position updating. In compared algorithms, invalid particles need to be reinitialized.

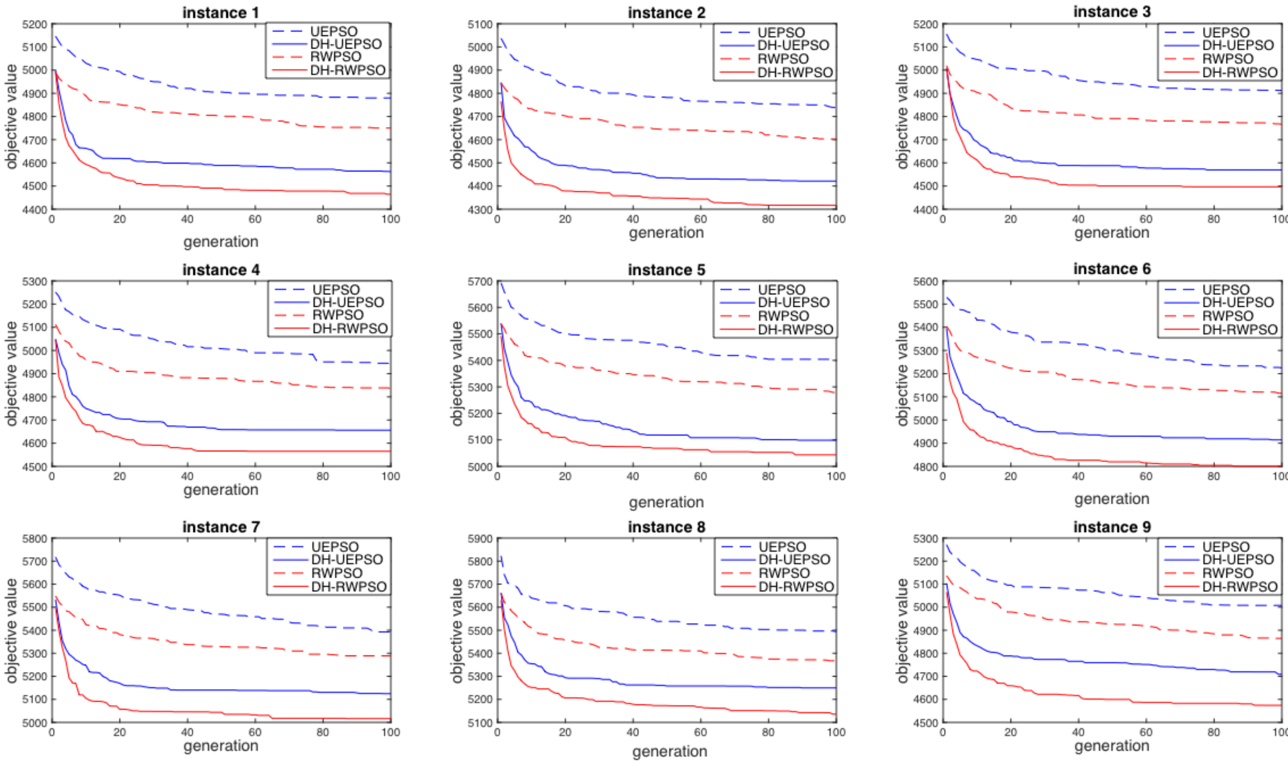

Fig.3. The converging curves of compared algorithms in different instances.

Hence, the number of invalid particles generated in iteration reflects the searching efficiency of PSO

We compare the invalid particles generate by all compared algorithms, which are concluded in Table II. The column "ratio" represents the ratio of invalid particles in two compared algorithms. We can observe that DH-PSO based approaches generated much less invalid particle (nearly 1/3) than original approaches during iterations. This is due to the one-stage mechanism and step-by-step position updating in DH-PSO. Every time a virtual node is mapped, the mapping of adjacent virtual links is also found, and the feasibility can be checked during solution construction. In DH-PSO, a virtual node is allowed to be mapped twice if substrate resources become congested. This mechanism is also helpful for finding feasible solutions. Another interesting phenomenon is that the invalid particles in RWPSO are less than those in UEPSO. This result is consistent with the performance in Fig. 3.

TABLE II
Comparison Results of Invalid Solutions for Compared Algorithms

	invalid particles					
	DH-UEPSO	UEPSO	ratio	DH-RWPSO	RWPSO	ratio
ins01	98.7	529.2	18.7%	74.8	299.4	25.0%
ins02	74.1	426.9	17.4%	52.4	215.6	24.3%
ins03	143.1	922.3	15.5%	104.6	457.6	22.9%
ins04	258.5	1198.5	21.6%	189.4	677.4	28.0%
ins05	192.4	684.5	28.1%	140.8	391.5	36.0%
ins06	136.9	514.3	26.6%	93.0	269.5	34.5%
ins07	253.8	1123.0	22.6%	183.7	565.1	32.5%
ins08	476.3	1476.2	32.3%	363.8	863.0	42.2%
ins09	142.4	542.0	26.3%	104.4	314.6	33.2%
ins10	107.1	476.7	22.5%	73.9	239.4	30.9%

V. CONCLUSION

VNE is the key technology in network virtualization to overcome Internet ossification. In this paper, the one-stage and dual-heuristic particle swarm optimization (DH-PSO) is devised to solve VNE. DH-PSO has two contributions. Firstly, DH-PSO updates positions step by step and thus the node mapping and link mapping are coordinated in one stage. Secondly, the dual-heuristic strategy is devised to further improve the performance of DH-PSO. DH-PSO can be combined with different two-stage PSO to become one-stage. In experimental studies, DH-PSO can generate less invalid solutions due to the one-stage mechanism and significantly outperform two-stage approaches.

For future researches, we will extend the dual-heuristic strategy and the step-by-step mechanism to other evolutionary algorithms, such genetic algorithms and estimation of distribution algorithms. Also, these are a lack of distributed approaches to VNE. It is promising to combine evolutionary algorithms with distributed computing to solve large-scale VNE problems.

REFERENCES

[1] A. Fischer, J. F. Botero, M. Till Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Communications Surveys & Tutorials,* vol. 15, no. 4, pp. 1888-1906, 2013.
[2] N. M. K. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," *IEEE Commun. Mag.,* vol. 47, no. 7, 2009.
[3] N. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Comput. Netw.,* vol. 54, no. 5, pp. 862-876, 2010.
[4] Y. Jin and Y. Wen, "When cloud media meet network function virtualization: challenges and applications," *IEEE Multimedia,* vol. 24, no. 3, pp. 72-82, 2017.

[5] Q. Duan, Y. Yan and A. V. Vasilakos, “A survey on service-oriented network virtualization toward convergence of networking and cloud computing,” *IEEE Trans. Netw. Service Manag.,* vol. 9, no. 4, pp. 373-392, 2012.
[6] J. Lischka and H. Karl, “A virtual network mapping algorithm based on subgraph isomorphism detection,” *Proceedings of the 1st ACM workshop on VISA'09*, 2009, pp. 81-88.
[7] L. Hong, L. Zou, X. Lian, and P. S. Yu, “Subgraph matching with set similarity in a large graph database,” *IEEE T. Knowl. Data En.,* vol. 27, no. 9, pp. 2507-2521, 2015.
[8] M. Chowdhury, M. R. Rahman and R. Boutaba, “Vineyard: Virtual network embedding algorithms with coordinated node and link mapping,” *IEEE/ACM Trans. Netw.,* vol. 20, no. 1, pp. 206-219, 2012.
[9] I. Fajjari, N. A. Saadi, G. Pujolle, and H. Zimmermann, “VNE-AC: Virtual network embedding algorithm based on ant colony metaheuristic,” *IEEE ICC*, 2011, pp. 1-6.
[10] S. Qing, J. Liao, J. Wang, X. Zhu, and Q. Qi, “Hybrid virtual network embedding with K-core decomposition and time-oriented priority,”, 2012, pp. 2695-2699.
[11] M. T. Beck, A. Fischer, H. de Meer, J. F. Botero, and X. Hesselbach, “A distributed, parallel, and generic virtual network embedding framework,”, 2013, pp. 3471-3475.
[12] S. Haeri and L. Trajkovi, “Virtual network embedding via monte carlo tree search,” *IEEE Trans. Cybern.,* vol. 48, no. 2, pp. 510-521, 2018.
[13] L. Gong, H. Jiang, Y. Wang, and Z. Zhu, “Novel location-constrained virtual network embedding (LC-VNE) algorithms towards integrated node and link mapping,” *IEEE/ACM Trans. Netw.,* vol. 24, no. 6, pp. 3648-3661, 2016.
[14] J. Kenndy and R. C. Eberhart, “Particle swarm optimization,” *Proc. IEEE Int. Conf. Neural Netw.,* pp. 1942-1948, 1995.
[15] Z. Zhang, X. Cheng, S. Su, Y. Wang, K. Shuang, and Y. Luo, “A unified enhanced particle swarm optimization - based virtual network embedding algorithm,” *Int. J. Commun. Syst.,* vol. 26, no. 8, pp. 1054-1073, 2013.
[16] X. Cheng, S. Su, Z. Zhang, K. Shuang, F. Yang, Y. Luo, and J. Wang, “Virtual network embedding through topology awareness and optimization,” *Comput. Netw.,* vol. 56, no. 6, pp. 1797-1813, 2012.
[17] A. Song, W. N. Chen, T. Gu, H. Yuan, S. Kwong, and J. Zhang, “Distributed virtual network embedding system with historical archives and set-based particle swarm optimization,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* pp. 1-16, in press, 2019.
[18] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang, “Virtual network embedding through topology-aware node ranking,” *ACM SIGCOMM Comp. Com.,* vol. 41, no. 2, pp. 38-47, 2011.
[19] N. M. Chowdhury, M. R. Rahman and R. Boutaba, “Virtual network embedding with coordinated node and link mapping,” *INFOCOM*, 2009, pp. 783-791.
[20] D. Palanikkumar and S. Priya, “Ant colony based graph theory (ACGT) and resource virtual network mapping (RVNM) algorithm for home healthcare system in cloud environment,” *Multimed. Tools Appl.,* 2018.
[21] W. Chen, D. Tan, Q. Yang, T. Gu, and J. Zhang, “Ant colony optimization for the control of pollutant spreading on social networks,” *IEEE Trans. Cybern.,* pp. 1-13, 2019.
[22] A. Song, W. N. Chen, Y. Gong, X. Luo, and J. Zhang, “A Divide-and-conquer evolutionary algorithm for large-scale virtual network embedding,” *IEEE T. Evolut. Comput.,* pp. 1-15, 2019.
[23] H. Hamedmoghadam., M. Jalili., P. Moradi., and X. Yu., “A global optimization approach based on opinion formation in complex networks,” *IEEE Transactions on Network Science and Engineering,* pp. 1-15, early access, 2018.
[24] W. Chen, Y. Jia, F. Zhao, X. Luo, X. Jia, and J. Zhang, “A cooperative co-evolutionary approach to large-scale multisource water distribution network optimization,” *IEEE Trans. Evol. Comput.,* pp. 1-15, early access, 2019.
[25] T. Zhao, W. Chen, A. W. Liew, T. Gu, X. Wu, and J. Zhang, “A binary particle swarm optimizer with priority planning and hierarchical learning for networked epidemic control,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* pp. 1-15, 2019.
[26] Q. Yang, W. Chen, T. Gu, H. Zhang, H. Yuan, S. Kwong, and J. Zhang, “A distributed swarm optimizer with adaptive communication for large-scale optimization,” *IEEE Trans. on Cybern.,* pp. 1-16, 2019.
[27] S. Su, Z. Zhang, A. X. Liu, X. Cheng, Y. Wang, and X. Zhao, “Energy-aware virtual network embedding,” *IEEE/ACM Trans. Networking,* vol. 22, no. 5, pp. 1607-1620, 2014.
[28] E. W. Zegura, K. L. Calvert and S. B. Acharjee, “How to model an internetwork,” *Proc. 15th Annu. IEEE INFOCOM*, 1996, pp. 594-602.